所属类别	**2020 年“华数杯”全国大学生数学建模竞赛**	参赛编号

脱贫帮扶绩效评价体系的构建与探究

摘要

2015 年，国家为了更好的开展扶贫工作，激励各帮扶单位提高扶贫效率，启动了脱贫帮扶绩效评价机制。本文运用附件提供的已经标准化处理过的数据，分析了五个评价指标的对应关系，研究脱贫帮扶绩效的评价和帮扶单位在单项评价指标上的帮扶业绩，并探索出影响获得“脱贫先进村庄”荣誉称号的重要因素。

针对问题一，我们考虑到指标评分在空间、时间方面的联系，设置了**四个切入角度**，采用**斯皮尔曼相关系数**分析各评价指标的对应关系。发现五年前的评分与 2020 年对应的指标评分有高度正相关的直接关联，此外 2015 年与 2020 年的各项指标之间均具有较高正相关性，同时五年来各指标之间的相关程度均没有发生改变。

针对问题二，我们围绕帮扶单位扶贫时存在的多方面差异构建了**脱贫帮扶绩效评价体系**。通过探究村庄集各指标评分的进步幅度、参考**局部均衡分析理论**研究均衡发展应付出的代价、采用**数据包络分析**定量评价帮扶难度不同对帮扶效率的影响，建立了**多属性决策排序模型**，最终确定 2020 年各项指标评分、2020 年总得分、五年来总得分的排名变化、各项指标评分的排名变化、均衡代价和帮扶效率为绩效评定指标。利用**因子分析法降维处理**各评定指标，求解得到了脱贫帮扶绩效的综合评价指数(2-8)。最后计算出所有单位的绩效得分，对不同类型的帮扶单位进行绩效排序，得到绩效较高的单位类型为 0、1，给出绩效前十名的单位编号为 33、1、0、21、2、22、11、56、25。

针对问题三，基于第二问建立的多属性决策排序模型，将多属性评价转变为针对各单项评价指标进行帮扶业绩评定的**单属性评价**，选取各村庄集五年来的各个指标评分的排名变化和帮扶工作的难度系数作为业绩评定指标，构建帮扶单位在各单项指标上的**帮扶业绩评价体系，并**引入 **BCG 矩阵**对体系的评价标准进行解释。通过计算业绩评定指标的**变异系数**确定各指标权重以降低主观性，再运用 **TOPSIS 法**求解帮扶业绩评定指数(3-9)，计算出所有帮扶单位分别在五个评价指标上的帮扶业绩，最后给出各单项评价指标中排名前五的帮扶单位编号（见表 3-1）。

针对问题四，对于 2020 年评分数据被删除的这十个村庄，首先我们对原始数据采取**特征工程**来提取和创造特征，为数据的预测提供条件。然后建立 **XGBoost 模型**和**随机森林模型**，通过对比预测的方式预测出 2020 年的五项指标评分缺失值，检验所得预测评分的可靠性；再通过构建**单层神经网络**做 **LinearRegression** 预测 2020 年的总分缺失值。选定影响获取“脱贫先进村庄”荣誉称号的重要因素是 2020 年总分排名和五年来各项指标评分的进步幅度，对所有村庄进行排序以得到排名前 10000 名的村庄编号，观察排序结果，找到这十个村庄中符合评选条件的有编号为 25149、39257、47883、12722、34276、12916 的村庄，但这十个村庄均不能入选一级“脱贫先进村庄”。

最后依据上述研究成果，给国家扶贫办写一封信阐述我们的观点和建议。

关键词：斯皮尔曼相关系数、因子分析、变异系数、随机森林模型、XGBoost 模型

一、问题重述

1.1 背景阐述

党的十八大以来，国家把扶贫开发工作纳入“四个全面”战略布局，作为实现第一个百年奋斗目标的重点工作。为了更好的激励各帮扶单位提高扶贫效率，五年前，国家启动了脱贫帮扶绩效评价机制。

某科研团队对全国 32165 个需要帮扶的贫困村进行了初步贫困调查，从居民收入（SR）、产业发展（CY）、居住环境（HJ）、文化教育（WJ）、基础设施（SS）等五个评价指标给出了评分。以此为依据划分被帮扶的村庄为 160 个集合，每个集合指定帮扶单位（标记为 0-159）进行帮扶，按照单位属性（如国企还是民营企业等）划分这 160 个帮扶单位为 6 个类型（标记为 0-5）。2020 年，研究团队再次进行调研，得到了被帮扶的这些村庄在五个评价指标方面的评分数据以及总分数据。所有数据都进行了标准化处理（标准化后的数值越大表示评分越高）。

160 个帮扶单位帮扶着基础不同的村庄，帮扶工作的态度、目标、投入、帮扶干部素质等是有差异的。为了保证评判的公正性，脱贫帮扶绩效的评价不能仅以最后各村庄的评价得分作为依据，还需要考虑各个评价指标的进步幅度。

1.2 问题提出

现根据上述题目背景及附件所提供的数据，建立数学建模解决以下问题：

问题一：判断五年前的评分与 2020 年对应的各项评分之间是否有直接的关联，分析各个评价指标的对应关系。

问题二：运用附件数据，阐明什么类型的帮扶单位、哪些帮扶单位在脱贫帮扶上有较高的绩效。为不同类型的帮扶单位绩效排序，并给出绩效前十名的帮扶单位编号。

问题三：每个帮扶单位在扶贫上有不同的工作特色，分析哪些帮扶单位分别在居民收入、产业发展、居住环境、文化教育、基础设施等评价指标上的帮扶业绩明显，列出各单项评价指标前五名的帮扶单位编号。

问题四：全国计划给予 10000 个村庄“脱贫先进村庄”称号，探索哪些因素对获得这个荣誉称号有着非常重要的影响。数据表中最后 10 个村庄的 2020 年评价分数被删除，判断他们能否评上“脱贫先进村庄”称号。如果称号分为一级和二级（一、二级称号比例为 1:3），判断这 10 个村庄中谁能评上“脱贫先进一级村庄”称号。

问题五：依据上述研究成果，给国家扶贫办写一封信，阐述观点和建议。

二、问题分析

2.1 问题一的分析

题目要求分析各个评价指标的对应关系，由于附件提供的指标评分标准化数据不服从正态分布，所以本文通过建立斯皮尔曼相关系数模型探究各个指标的相关性，判断本文中五年前的评分与 2020 年对应的指标评分是否存在着直接关联。考虑到指标评分在空间、时间方面的联系，设置了四个切入角度对各指标之间的对应关系分别进行空间角度分析、时间角度分析、时空角度综合分析。

2.2 问题二的分析

构建一个好的脱贫帮扶绩效评价体系对各帮扶单位的绩效得分进行评价，需要考虑到帮扶单位开展扶贫工作时各方面存在的差异、每个村庄集合的基础不同导致帮扶难度的不一致以及可能出现帮扶效果极端化的现象。我们知道仅以 2020 年各村庄的评分高低作为脱贫帮扶绩效的评价依据显然是不够全面且不具有代表性和说服力的。因此本文通过建立多属性决策排序模型，选定绩效评定指标。利用因子分析法对绩效评定指标进行降维处理，然后根据脱贫帮扶绩效评价体系，利用求解得到的各帮扶单位绩效评价指数算出各个单位的脱贫帮扶绩效得分。最后对不同类型的帮扶单位进行绩效排序，得到绩效较高的单位类型并给出绩效前十名的帮扶单位编号。

2.3 问题三的分析

不同于第二问，第三问需要对帮扶单位在各单项评价指标方面的帮扶业绩进行评价。因此在第二问建立的多属性决策排序模型的基础上，将多属性评价转变为单属性评价，每次只针对一个指标来评价帮扶单位的业绩。着重考虑各村庄集合五年来单项指标评分的变动情况、帮扶单位开展工作的难度，构建单项指标方面的帮扶业绩评价体系，选定业绩评定指标并引入 BCG 矩阵明确业绩明显的评价标准。利用 TOPSIS 法求解评价模型，评价各帮扶单位在单项指标方面的帮扶业绩，为了减少主观性和消除评定指标量纲的影响，本文通过计算各项评定指标的变异系数来确定 TOPSIS 算法中指标的权重。最终得到各单项评价指标方面帮扶业绩排名前五的帮扶单位编号。

2.4 问题四的分析

由于数据表中有十个村庄的 2020 年的评价分数被删除，因此我们首先需要预测出这十个村庄的缺失数据。为了降低预测模型的泛化误差，对数据建立特征工程，将原始数据转换为更能代表预测模型潜在问题的特征，通过挑选最相关的特征，提取特征以及创造特征。然后建立 XGBoost 模型和随机森林模型分别对缺失的 2020 年各指标评分数据进行预测，通过对比检验预测数据的可靠性；再构建单层神经网络 LinearRegression 模型预测十个村庄的 2020 年总分数据。根据我们所确定的影响获得“脱贫先进村庄”荣誉称号的重要因素对各个村庄进行相关排序，得到排名前 10000 名的村庄编号，观察排序情况，找出这十个村庄中符合评选条件的并记录编号。

三、问题假设

1、假设附件提供的关于 32165 个村庄在 2015 年与 32155 个村庄在 2020 年的各项指标评分数据和总分数据真实可信。

2、假设 2015 年的打分考核标准与 2020 年的打分考核标准相同。

3、假设帮扶单位对所帮扶的村庄不会因主观喜好来改变帮扶资源分配。

4、不考虑自然灾害等小概率事件在 2015 到 2020 年对评价对象的绩效、业绩评分造成影响。

5、假设不存在帮扶资源的浪费，即帮扶资源都得到有效利用。

四、符号说明

符号	说明
$P_{t,i,j}$	在年份t第j个村庄集的第i项评价指标评分排名
$\Delta P_{i,j}$	第j个村庄集的第i个指标评分在五年内的排名变化情况
$\overline{\Delta P_j}$	第j个村庄集五年来的评价总分排名变化情况
$Cost_j$	第j个村庄集均衡发展的代价
C_i	第i个帮扶单位的帮扶村庄数量
N_i	第i个帮扶单位扶贫工作的难度系数
SR_t	居民收入在年份t的评分
CY_t	产业发展在年份t的评分
HJ_t	居住环境在年份t的评分
WJ_t	文化教育在年份t的评分
SS_t	基础设施在年份t的评分
S_t	年份t的综合评价总分

五、模型的建立与求解

5.1 问题一模型的建立与求解

常用来检验指标之间相关关系的方法有皮尔逊相关系数、斯皮尔曼相关系数，我们对样本数据序列进行 JB 检验，发现样本的指标评分标准化数据不服从正态分布，所以不能使用皮尔逊相关系数进行分析。本文考虑使用斯皮尔曼相关系数来探究各指标的对应关系，通过定量分析得到相对可靠的结果。

5.1.1 确定分析各个评价指标对应关系的切入角度

脱贫帮扶绩效评价机制从居民收入 SR、产业发展 CY、居住环境 HJ、文化教育 WJ、基础设施 SS 等五个评价指标给出了评分。题目首先要求我们判断五年前的评分与 2020 年对应的各项评分之间是否有直接的关联，因此本文考虑从①2015 年的指标评分与 2020 年对应的指标评分之间，②2015 年的各指标评分之间，③2020 年的各指标评分之间，④2015 年与 2020 年各指标评分之间这四个切入角度来分析各个评价指标的对应关系。

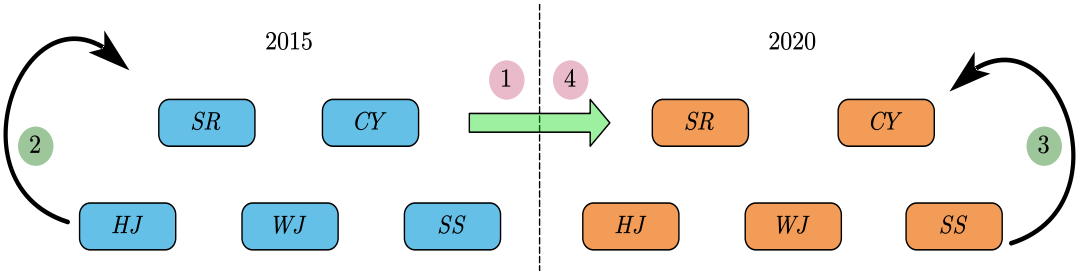

图 1-1 分析对应关系的四个切入角度

其中切入角度②、③属于空间角度分析，角度①为时间角度的分析，而角度④是时空角度的综合分析。从空间和时间两个角度切入对各个评价指标的对应关系进行全方面探究，有利于提高分析结果的准确性和有效性。

5.1.2 建立斯皮尔曼相关系数模型

考虑到斯皮尔曼相关系数是用来描述两个定序数据集之间的相关性，所以我们首先对指标的评分大小进行排序，将本文的定量数值序列转化为定序数据序列，再计算各指标评分之间的斯皮尔曼相关系数，并检验相关系数的显著性[1]。任意两个数据集之间的斯皮尔曼相关系数的具体计算公式为：

$$r_s = 1 - \frac{6\sum_{i=1}^{n} d_i^2}{n(n^2-1)} \tag{1-1}$$

其中d_i表示两个指标评分之间的等级差。r_s的变化区间为[-1,1]，负值表示负相关，正值表示正相关，其绝对值越接近 1 代表两个评价指标的相关性越强。本文认定r_s的绝对值大于 0.6 为高度相关，小于 0.4 为低度相关，介于二者之间为中度相关。

5.1.3 探究各个评价指标的对应关系

5.1.3.1 各项指标在时间角度、时空角度的对应关系

得到 2015 年与 2020 年各个指标评分之间的斯皮尔曼相关系数计算结果如下：

2020年 / 2015年	SR	CY	HJ	WJ	SS
SR	0.54013584	0.49962009	0.5900673	0.4549577	0.4638065
CY	0.49714222	0.65562948	0.59535136	0.59282634	0.56707429
HJ	0.58471389	0.60776127	0.80009998	0.55064221	0.56044723
WJ	0.52267268	0.66064017	0.6165194	0.67083428	0.63825762
SS	0.52928773	0.6631929	0.6265707	0.63408232	0.6357948

图 1-2　2015 年与 2020 年各指标评分之间的斯皮尔曼相关系数

观察上图可以发现 2015 年与 2020 年的各项指标之间均具有正相关性。

从角度①进行分析，产业发展 CY、居住环境 HJ、文化教育 WJ 和基础设施 SS 这四个评价指标在五年前的评分和该指标在 2020 年的对应评分具有高度正相关性。而指标居民收入 SR 的 2015 年评分和 2020 年评分间的相关性相对于其他几个指标来说较弱，因而不能绝对的说一个地方五年前的居民收入不错，现在的居民收入也会不错。这是因为存在部分村庄的居民收入水平在这五年来有明显提升，导致有些居民收入较好的村庄相比之下提升幅度较小。

下面对此规律进行深入探究。对 32165 个数据样本进行规律关联统计，选取总体数据中居民收入 SR 在 2015 年的评分排名前α% 的个体，并记录村庄编号；观察 2020 年 SR 评分排名在前α% 的个体中原来被记录的个体所占的比重β。

表 1-1 规律关联统计

α	10%	20%	30%	40%
β	34.12%	52.43%	57.41%	58.86%

根据上表，2015 年居民收入不错的村庄在 2020 年继续保持的概率不大。说明关联规律并不可信，由于 2020 年各项指标的评分不仅与五年前的对应评分高度相关，同时和 2015 年的其他指标评分也存在较强相关性，也就是说 2020 年的各项指标评分受到了 2015 年全五个指标所构成的系统的影响。

2020 年的各项指标评分除了和自身在 2015 年的对应评分高度相关外，从角度④分析对应关系，2020 年指标居民收入 SR、产业发展 CY 的评分都分别与 2015 年的居住环境 HJ、文化教育 WJ 和基础设施 SS 具有中度正相关性；2020 年指标 HJ 的评分与 2015 年 CY、SR 的评分中度正相关，和 WJ、SS 的评分高度正相关；2020 年指标 WJ 的评分与五年前的 CY、HJ 的评分中度正相关，和 SS 评分高度正相关；2020 年指标 SS 的评分与五年前 CY、HJ 的评分中度正相关，和 WJ 评分高度正相关。

5.1.3.2 各项指标在空间角度的对应关系

分别得到 2015 年、2020 年的各项指标评分之间的斯皮尔曼相关系数计算结果如下：

2015年 \ 2015年	SR	CY	HJ	WJ	SS
SR	1	0.5386874	0.6240967	0.5400264	0.5536797
CY	0.5386874	1	0.6302759	0.6875103	0.6856059
HJ	0.6240967	0.6302759	1	0.6396631	0.659824
WJ	0.5400264	0.6875103	0.6396631	1	0.7469397
SS	0.5536797	0.6856059	0.659824	0.7469397	1

(a) 2015 年的各项指标评分之间

2020年 \ 2020年	SR	CY	HJ	WJ	SS
SR	1	0.5821486	0.6486684	0.5487132	0.559258
CY	0.5821486	1	0.6881018	0.7638204	0.73845
HJ	0.6486684	0.6881018	1	0.6641654	0.6723548
WJ	0.5487132	0.7638204	0.6641654	1	0.7528583
SS	0.559258	0.73845	0.6723548	0.7528583	1

(b) 2020 年的各项指标评分之间

图 1-3 各项指标评分之间的斯皮尔曼相关系数

根据图 3 可知 2020 年各项指标之间相关程度的分布与 2015 年保存一致，说明 2020 年和五年前各项指标评分的相关程度和结构没有发生改变。除了时间角度的对应关系外，空间里各项指标之间的对应关系为：居民收入 SR 仅与居住环境 HJ 有高度正相关性；产业发展 CY 与居住环境 HJ、文化教育 WJ、基础设施 SS 之间都是高度正相关；居住环境 HJ 与其他四个指标均具有高度正相关性；文化教育 WJ、基础设施 SS 和除居民收入 SR 之外的三个指标均高度正相关。

以上计算出的各个指标间的斯皮尔曼相关系数均通过了显著检验，且在 0.01 的置信水平下具有显著性。

5.2 问题二脱贫帮扶绩效评价体系的建立与求解

5.2.1 脱贫帮扶绩效评定指标的选取

由于 160 个帮扶单位帮扶的村庄基础不同，同时帮扶单位开展帮扶工作时在各方面存在差异，仅用 2020 年各村庄的评分高低无法真正有效的体现一个帮扶单位在脱贫攻坚提升方面所做出的努力。为了保证评判帮扶效果的公正性、客观性和有效性，我们认

为一个好的脱贫帮扶绩效评价体系应该兼具以下几点：

(1) 避免只依据单一的评分数值与总分排名

不只依靠 2020 年被帮扶村庄所得总分的排名和各指标评分来确定帮扶绩效名次，同时还要考虑总分与五年前相比排名的差异和各个指标评分的进步幅度。如：有的被帮扶村庄虽然各项指标评分与总分排名不是很高，但相较 2015 年的得分已经有了较大的提升；虽然有的村庄各指标评分与总分排名很高，但相较五年前得分水平没有提升甚至出现倒退。那么前者的帮扶绩效就应该高于后者。

(2) 避免单位帮扶工作的帮扶效果极端化

【被帮扶个体的帮扶效果极端化】：帮扶单位的帮扶策略可以根据不同的帮扶对象做出变化，最后的帮扶效果指被一个单位帮扶的所有村庄达成的总体脱贫提升结果。如：在一个单位的帮扶下，只有少数几个村庄有较大的脱贫提升，而其余的村庄脱贫提升不模型甚至贫困程度加深，这就说明发生了被帮扶个体的帮扶效果极端化的现象。

【指标评分的帮扶效果极端化】：一个优秀的帮扶绩效体现在各项指标评分的进步，在帮扶的过程中或多或少会出现各项指标评分没有全方面进步、进步幅度不均匀甚至某些指标评分降低的情况，说明发生了指标评分的帮扶效果极端化的现象。

为了避免以上情况的发生就要确保单位的帮扶工作全方面开展，做到帮扶效果发展均衡，因此在评价脱贫帮扶绩效时需要将效果均衡考虑进去。

(3) 考虑到帮扶的难度不一致对帮扶效率的影响

不仅要考虑不同单位所帮扶村庄的进步，还要充分考虑帮扶单位开展帮扶工作的难度，保证帮扶绩效评价更全面、更准确有效。

【要帮扶的村庄数量不同】：不同的集团的帮扶数量不同。帮扶对象的数量越多说明帮扶的难度越大，帮扶对象的数量越少说明帮扶的难度越小。

【被帮扶村庄的基础不同】：某个单位帮扶的村庄集合的原始基础相较于其他村庄集更差，如果要求总分排名进步相同的名次，相对于其他集合来说该村庄集合的提升难度大，即该村庄集合的脱贫提升相对困难。

综上，本文最终选取 2020 年各项指标评分、2020 年总得分、五年来总得分的排名变化、各项指标评分的排名变化,、均衡代价和帮扶效率作为绩效评定指标。

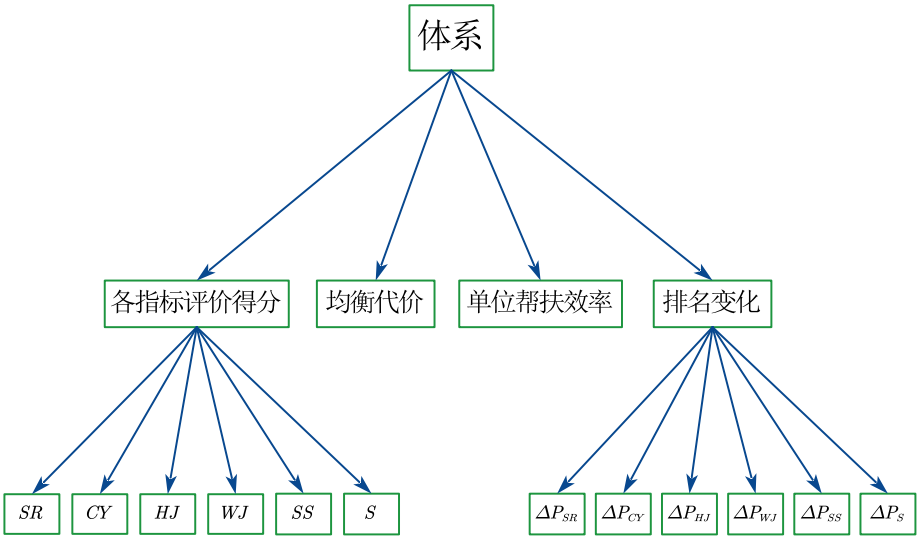

图 2-1 脱贫帮扶绩效评价体系

5.2.2 建立多属性决策排序模型

5.2.2.1 每个村庄集的各项指标评分的进步幅度

题中附件提供的数据已经经过标准化处理，说明 2015 年各个指标评分数据的相对大小与排名不变，但无法与 2020 年标准化后的样本数据做比较，因此本文利用五年来五个指标评分的排名变化情况表示各项指标评分从 2015 年-2020 年的进步幅度。假设在年份t第j个样本集的第i个指标评分排名为$P_{t,i,j}$，第j个样本集的第i个指标评分在五年内的排名变化情况为$\Delta P_{i,j}$，则有：

$$\Delta P_{i,j} = P_{2020,i,j} - P_{2015,i,j} \tag{2-1}$$

5.2.2.2 均衡帮扶，避免帮扶效果极端化

关于均衡帮扶的度量，参考经济学中的局部均衡分析理论（引用）来分析每个样本集达到帮扶效果均衡的代价，设第j个样本集五年来的评价总分排名变化为$\overline{\Delta P_j}$，则：

$$\overline{\Delta P_j} = \frac{\Delta P_{SR,j} + \Delta P_{CY,j} + \Delta P_{HJ,j} + \Delta P_{WJ,j} + \Delta P_{SS,j}}{5} \tag{2-2}$$

而第j个样本集要达到均衡发展付出的代价$Cost_j$为：

$$Cost_j = \left(\Delta P_{SR,j} - \overline{\Delta P_j}\right)^2 + \left(\Delta P_{CY,j} - \overline{\Delta P_j}\right)^2 + \left(\Delta P_{HJ,j} - \overline{\Delta P_j}\right)^2 + \left(\Delta P_{WJ,j} - \overline{\Delta P_j}\right)^2 + \left(\Delta P_{SS,j} - \overline{\Delta P_j}\right)^2 \tag{2-3}$$

5.2.2.3 帮扶难度不一致对帮扶效率的影响

利用数据包络分析 DEA 中的C^2R法定量评定各帮扶单位扶贫工作的帮扶效率，使用C^2R法能够充分考虑到由于每个单位的帮扶工作难度不一致对帮扶效率造成的影响。

Step 1. 建立数据包络--绩效评价模型

设有n个决策单元（DMU），每个决策单元有a种输出和b种输入。输入向量为$X_i = (x_{1i}, x_{2i}, ..., x_{bi})^T (i = 1, 2, ..., n)$，输出向量为$Y_j = (y_{1j}, y_{2j}, ..., y_{aj})^T (j = 1, 2, ..., n)$。$V = (v_1, v_2, ..., v_a)^T$为输出的权重向量，$W = (w_1, w_2, ..., w_b)^T$为输入的权重向量。第$m$个决策单元的效率评价函数为

$$h_m = \frac{W^T Y_m}{V^T X_m} (m = 1, 2, \cdots, n) \tag{2-4}$$

则第m个决策单元的相对效率优化评价模型为

$$\max \frac{W^T Y_m}{V^T X_m} \tag{2-5}$$

其中$s.t.\begin{cases} \dfrac{W^T Y_i}{V^T X_i} \leq 1 \quad i = 1, 2, ..., n \\ V \geqslant 0 \\ W \geqslant 0 \end{cases}$

将分式规划通过 Charnes-Cooper 变换化为线性规划求解，令$\omega = tV$，$\mu = tW$，$t = \dfrac{1}{V^T X_m}$，代入公式(2-5)可得：

$$\max V_m = \mu^T Y_m \tag{2-6}$$

其中$s.t.\begin{cases}\omega^T X_i-\mu^T Y_i\geqslant 0,\ \ i=1,2,...,n\\ \omega^T X_m=1\\ \omega\geqslant 0,\mu\geqslant 0\end{cases}$

通过观察V_m的最大值可以得到在考虑了每个单位帮扶难度不一致的情况下脱贫帮扶绩效的定量化评价得分。

Step 2. 求解数据包络--绩效评价模型

本题有 160 个决策单元，每个帮扶单位相当于一个决策单元，该决策单元的输入为每个单位所帮扶的村庄数目的倒数，设第m个单位的帮扶村庄数为C_i，则输入$X_m=\left[\frac{1}{C_m}\right]^T$，输出$Y_j=[\Delta P_{SR,j},\Delta P_{CY,j},\Delta P_{HJ,j},\Delta P_{WJ,j},\Delta P_{SS,j}]^T$，在考虑帮扶难度不一致的情况下的第$m$个单位的帮扶效率得分为$V_i$。

5.2.3 因子分析法求解模型

Step 1. 求解各绩效评定指标之间的相关系数矩阵

利用标准化处理之后的各村庄集合的 2020 年总得分S_{2020}、五年来的总得分排名变化$\Delta P^{(Z)}$、2020 年各项指标评分$SR_{2020},CY_{2020},HJ_{2020},WJ_{2020},SS_{2020}$、各项指标的评分排名变化$\Delta P_{SR}{}^{(Z)},\Delta P_{CY}{}^{(Z)},\Delta P_{HJ}{}^{(Z)},\Delta P_{WJ}{}^{(Z)},\Delta P_{SS}{}^{(Z)}$、均衡代价$Cost^{(Z)}$、帮扶效率$V^{(Z)}$的样本数据，求解各评定指标之间的相关系数，相关系数大小实际上反映的是公因子起作用的空间[3]。计算标准化数据间相关系数的公式为：

$$r_{ij}=\frac{\sum_{i=1}^{n}X_{ti}\cdot X_{tj}}{n-1}\qquad(i,j=1,2,\cdots,m)\tag{2-7}$$

其中r_{ij}代表指标i与指标j之间的相关系数。相关系数矩阵为$R=(r_{ij})_{m\times m}$，详见附录。

Step 2. KMO 和 Bartlett 球形度检验

接下来对各指标变量进行 KMO 和 Bartlett 球形度检验，判断变量是否适合进行因子分析。其中 KMO 检验是对原始变量之间的相关系数大小进行检验；Bartlett 球形检验是检验各个变量之间的相关性程度，指标之间的相关性越强，说明越适合做因子分析。

表 2-1　KMO 和 Bartlett 球形度检验结果

KMO 取样适切性量数		0.779
Bartlett 球形度检验	近似卡方	4009.601
	自由度	91
	显著性（P 值）	0.000

由检验结果可知 KMO 值等于 0.779，且 Bartlett 球形度检验的 P 值=0.000<0.05，所以在 95%的置信水平下拒绝相关系数矩阵是一个单位矩阵的原假设，说明各指标变量之间存在相关性，表明各个评定指标数据较适合进行因子分析，将原来的指标转化为公共因子的效果较好。

Step 3. 主成分法、碎石图分析提取初等公因子

用主成分法计算初等因子载荷矩阵，确定初等因子。计算相关系数矩阵的特征值$\lambda_1\geqslant\lambda_2\geqslant\cdots\geqslant\lambda_m\geqslant 0$，求取相应的特征向量$(k_1,k_2,\cdots,k_m)$，则初等因子的载荷矩阵为

$A=(\sqrt{\lambda_1}k_1,\sqrt{\lambda_2}k_2,...,\sqrt{\lambda_m}k_m)$，得到 14 个初等因子($Z_1$，$Z_2$，…，$Z_{14}$)。根据碎石检验观察特征值的变化，确定选取的初等公因子数：

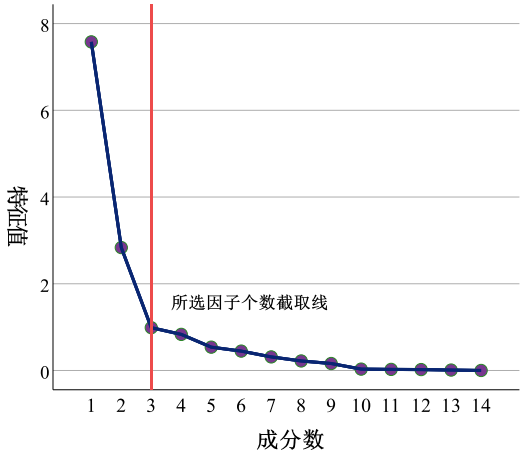

图 2-2 碎石图

参照碎石图，当成分数为 3 时特征值较前一特征值出现较大下降，并且该特征值数值较小，此后的特征值变化曲线趋于平缓，所以可以大致确定公因子的个数为 3。计算所有因子的方差贡献率及累积贡献率（见附录），可知前三个因子的累积贡献率达到了 81.426%>80%，即降维后得到的三个公因子概况了 80%以上的原始指标信息，具有较强代表性和有效性，符合公因子提取条件。结合碎石图分析结果确定最终选取的公因子个数为 3：Z_1、Z_2、Z_3。

Step 4. 最大方差法旋转因子载荷矩阵，进行降维分析

得到初等因子模型后，由于其中的公因子不一定可以反映出问题的实质特征，需要通过因子旋转使每个公因子上的载荷分配更清晰，从而减少解释公因子实际意义的主观性。利用最大方差法旋转因子，得到旋转后的载荷矩阵：

表 2-2 旋转后的因子载荷矩阵

原始指标	公因子		
	Factor 1	Factor 2	Factor 3
$\Delta P_{SR}^{(Z)}$	0.059	0.790	-0.129
$\Delta P_{CY}^{(Z)}$	-0.053	0.897	0.042
$\Delta P_{HJ}^{(Z)}$	0.216	0.871	0.066
$\Delta P_{WJ}^{(Z)}$	0.230	0.747	0.075
$\Delta P_{SS}^{(Z)}$	0.246	0.871	0.031
$\Delta P^{(Z)}$	0.584	0.594	-0.132
$Cost^{(Z)}$	0.462	-0.133	0.022
$V^{(Z)}$	0.129	0.009	0.978
SR_{2020}	0.942	0.235	0.096
CY_{2020}	0.957	0.218	0.057
HJ_{2020}	0.960	0.213	0.030
WJ_{2020}	0.937	0.267	0.049
SS_{2020}	0.960	0.182	0.075
S_{2020}	0.966	0.229	0.057

上表数据表示 3 个公因子与各原始指标变量之间的权重。在旋转后的公因子载荷矩阵中，选取荷重较大的绩效评定指标作为对应公因子的代表性指标。旋转后的因子图如下：

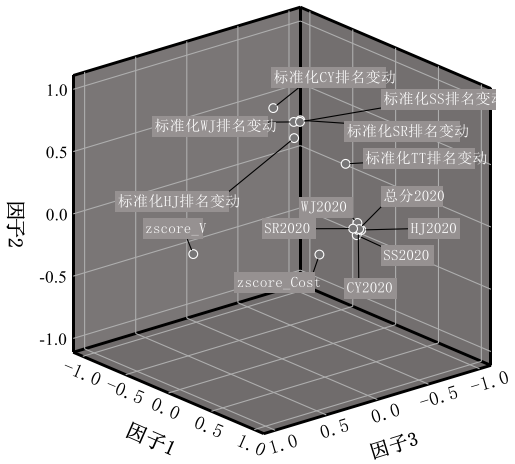

图 2-3 旋转后的因子图

参照旋转后的因子载荷矩阵与因子旋转图，对降维得到的三个公因子进行定义和解释：

(1) 第一公因子Z_1与 2020 年村庄集的五个评价指标评分SR_{2020},CY_{2020},HJ_{2020},WJ_{2020},SS_{2020}、2020 年总得分S_{2020}的相关性高，归纳为**评价得分因子**F_1；

(2) 第二公因子Z_2与五个评价指标的评分排名变化$\Delta P_{SR}^{(Z)}$,$\Delta P_{CY}^{(Z)}$,$\Delta P_{HJ}^{(Z)}$,$\Delta P_{WJ}^{(Z)}$,$\Delta P_{SS}^{(Z)}$、五年来总分排名变化$\Delta P^{(Z)}$的相关性较高，定义为**排名变化因子**F_2；

(3) 第三公因子Z_3与帮扶效率$V^{(Z)}$的相关性高，定义为**效率评分因子**F_3；

Step 5. 求解帮扶绩效的评价得分

我们将评价指标因子F_1、均衡代价因子F_2、居民收入因子F_3和帮扶效率因子F_4表示为原始指标变量的线性组合，以旋转后每个公因子的方差累积贡献率作为权重系数得到各因子的得分函数。

表 2-3 旋转后公因子的方差累积贡献率

原始指标	Factor 1	Factor 2	Factor 3
$\Delta P_{SR}^{(Z)}$	-0.059	0.221	-0.119
$\Delta P_{CY}^{(Z)}$	-0.105	0.267	0.063
$\Delta P_{HJ}^{(Z)}$	-0.050	0.233	0.066
$\Delta P_{WJ}^{(Z)}$	-0.035	0.196	0.071
$\Delta P_{SS}^{(Z)}$	-0.041	0.229	0.029
$\Delta P^{(Z)}$	0.068	0.111	-0.166
$Cost^{(Z)}$	0.105	-0.085	-0.016
$V^{(Z)}$	-0.050	0.008	0.972
SR_{2020}	0.159	-0.026	0.028
CY_{2020}	0.167	-0.033	-0.012
HJ_{2020}	0.170	-0.035	-0.040
WJ_{2020}	0.158	-0.017	-0.018

SS_{2020}	0.170	-0.043	0.005
S_{2020}	0.168	-0.031	-0.013

将每个公因子得分进行加权求和即可得到因子综合评价得分，即各帮扶单位的绩效评价指数为：

$$F = 0.543457F_1 + 0.366423F_2 + 0.09012F_3 \tag{2-8}$$

5.2.4 分析各帮扶单位的绩效得分

根据构造出的脱贫帮扶绩效评价体系，利用求解多属性决策排序模型得到的各单位绩效评价指数，计算出各个帮扶单位的绩效得分。给不同类型的帮扶单位绩效进行排序，得到脱贫帮扶绩效前十名的帮扶单位编号如下：

表 2-4 脱贫帮扶绩效前十名的单位排序

单位编号	F_1得分	F_2得分	F_3得分	绩效评价得分F	类型
33	2.0205	1.08859	-0.27875	1.471880176	0
1	1.96206	0.806	-0.13918	1.349155108	0
0	2.13853	0.46358	-0.26674	1.308108158	0
21	1.83241	0.5701	-0.25241	1.182052286	0
2	1.16446	1.47392	-0.0057	1.172414614	1
22	1.74304	0.55109	-0.1357	1.137032332	0
11	0.92315	1.49246	-0.0585	1.043297349	1
56	0.26507	2.42316	0.10064	1.040981046	1
25	1.50513	0.60412	-0.08784	1.031471582	1
99	1.62793	0.46824	-0.29859	1.02943416	1

根据上表可知，脱贫绩效评价得分高的帮扶单位类型均属于 0、1 类。计算出所有 6 种类型的帮扶单位的绩效平均得分，可以知道 0、1 类型的帮扶单位在脱贫帮扶上面有较高的绩效。

表 2-5 各种类型帮扶单位的绩效平均得分

单位类型	绩效平均分
0	0.84966978
1	0.552695703
2	-0.235690749
3	-0.226995482
4	-1.125044375
5	-0.516895261

5.3 问题三的建立与求解

5.3.1 构建各个帮扶单位在单项评价指标方面的帮扶业绩评价体系

5.3.1.1 帮扶业绩评定指标的选取

第三问的评价目标是帮扶单位在各单项评价指标方面的帮扶业绩，而不是脱贫帮扶

绩效，因此在第二问建立的多属性决策排序模型的基础上，将多属性评价转变为单属性评价，每次只针对一个指标来评价各单位的帮扶业绩。其中要着重考虑的因素有：

(1)各个村庄集合五年来该指标评分的变动情况

(2)帮扶单位开展工作的难度:基础好的村庄集提升一定的名次比基础差的村庄集提升相同的名次要容易，基础差的村庄集提升名次要超过原本基础好的村庄集是最为困难的，因为帮扶基础差的村庄要付出更多的资源、投入更多的精力；要求上升的名次越高，其难度越大。这个过程符合自然界中的逻辑斯蒂增长规律，逻辑斯蒂函数表达式为

$$f(x)=\frac{1}{1+e^{-x}} \tag{3-1}$$

被帮扶的第i个村庄集的难度系数为

$$N_i=f(P_{2020,i})-f(P_{2015,i}) \tag{3-2}$$

最终利用五年来村庄集的各项评价指标评分排名变化ΔP和帮扶工作难度系数N来评价各帮扶单位在各项评价指标方面的帮扶业绩。其中各项指标评分的排名变化是指2020年某一帮扶单位的单项评价指标排名与2015年该单位的同一项评价指标排名的差值。

5.3.1.2 帮扶业绩的评价标准

为了明确各帮扶单位在各个评价指标方面帮扶业绩明显的评价标准，我们引入波士顿（BCG）矩阵，直观解释这个评价标准[3]。

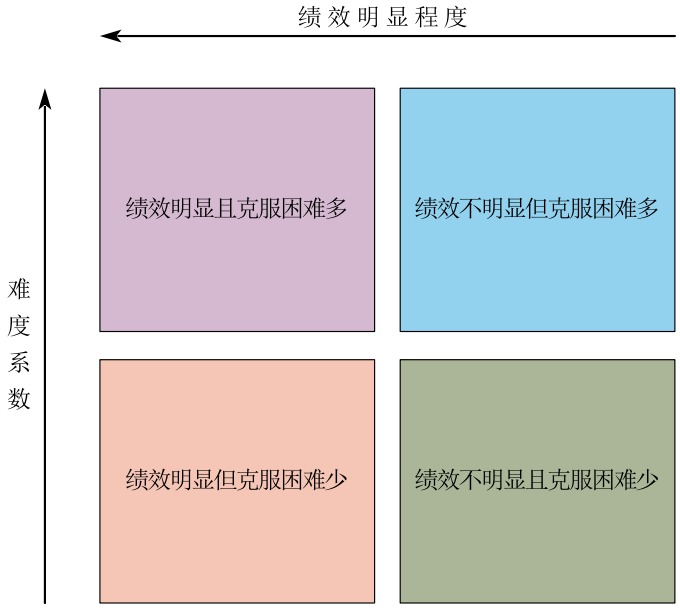

图 3-1 波士顿矩阵

如果将所有的单位分为四种类型，分别是“绩效明显且克服困难多”、“绩效小但克服困难多”、“绩效明显但克服困难少”、“绩效小且克服困难少”。而决定帮扶单位所属类型的是各个指标评分的排名变化和帮扶难度系数大小：单项指标评分的排名变化越大，代表帮扶单位达成的帮扶效果越好；帮扶难度系数越大，表示单位在帮扶工作开展过程中需要克服的困难越多。

5.3.2 TOPSIS 法求解帮扶业绩评价模型

TOPSIS 法（逼近理想解排序法）是系统工程中一种常见的有限方案多目标的决策分析法，能够充分利用原始数据的信息精确地反映出各个可行解方案之间的差距。TOPSIS 区别于层次分析法、模糊综合评价法，不需要目标函数，无需通过相应的检验，适用范围较广泛，但主观性较强[4]。

Step 1. 统一指标类型，计算评定指标权重

五年来村庄集的各项评价指标评分排名变化为极小型指标，则其标准化为：

$$\Delta P_i' = \frac{\max \Delta P - \Delta P_i}{\max \Delta P - \min \Delta P} \tag{3-3}$$

帮扶难度系数为极大型指标，则其标准化为：

$$N_i' = \frac{N_i - \min N}{\max N - \min N} \tag{3-4}$$

利用 TOPSIS 法进行帮扶业绩评价之前需要先确定业绩评定指标的权重。由于帮扶业绩评价体系中的两项评定指标量纲不同，不宜直接进行差距比较。为了消除量纲的影响，本文通过计算各项评定指标的变异系数来衡量指标取值之间的差异程度。具体的评定指标变异系数计算公式如下：

$$V_i = \frac{\sigma_i}{\overline{x}_i} \quad (i = 1, 2, \cdots, n) \tag{3-5}$$

其中V_i是第i项指标的变异系数，也称为标准差系数；σ_i是第i项指标的标准差；$\overline{x}_i$是第i项指标的平均数。则各业绩评定指标的权重计算公式为：

$$W_i = \frac{V_i}{\sum_{i=1}^{n} V_i} \tag{3-6}$$

Step 2. 确定最优解和最劣解并计算各评价对象到最优、劣解的距离。

根据本问构建的单项评价指标方面的帮扶业绩评价体系，计算各帮扶单位在单项评价指标方面的业绩评价与最优解之间的距离为

$$D_i^+ = \sqrt{\omega_1 (\max \Delta P - \Delta P_i)^2 + \omega_2 (\max N - N_i)^2} \tag{3-7}$$

各单位在单项评价指标方面的业绩评价与最劣解之间的距离为

$$D_i^- = \sqrt{\omega_1 (\min \Delta P - \Delta P_i)^2 + \omega_2 (\min N - N_i)^2} \tag{3-8}$$

Step 3. 获得各帮扶单位在单项评价指标方面的业绩评定值与最优值的相对接近程度，即单项指标方面的帮扶业绩评定指数：

$$D_i = \frac{D_i^-}{D_i^- + D_i^+} \tag{3-9}$$

5.3.3 结果分析

根据构造出的各个帮扶单位在单项评价指标方面的帮扶业绩评价体系，计算出所有帮扶单位分别在居民收入、产业发展、居住环境、文化教育、基础设施这五个评价指标上的帮扶业绩，给出各单项评价指标前五名的帮扶单位编号：

表 3-1 各单项评价指标帮扶业绩前五名的帮扶单位排序

排序	帮扶单位编号				
	居民收入	产业发展	居住环境	文化教育	基础设施
1	0	4	50	3	75
2	11	48	12	2	2
3	138	12	0	5	3
4	125	1	2	1	1
5	148	8	1	0	0

5.4 问题四模型的建立与求解

5.4.1 建立模型预测缺失数据

5.4.1.1 预测前的分析

题目要求判断被删除部分数据的十个村庄能否评上“脱贫先进村庄”称号，因此我们首先需要对被删除的数据进行准确预测。对于连续数据的预测，目前最常见的方式主要有多元回归分析、灰色预测、神经网络模型、支持向量机回归（SVC）等，但这些模型或多或少都存在着较大的缺陷。回归方程式的预测结果只是一种推测，而且受因子多样性和不可测性影响；灰色预测只能对非负数据进行预测，同时要求数据具有准指数规律。我们需要预测的是 2020 年的五项指标评分以及总得分，各个数据之间存在较为强烈的非线性。虽然神经网络模型和 SVC 也可以很好的拟合数据中的非线性，但由于本题的数据量较大，训练模型的耗时相对较长。集成学习作为强大的数据分析方法，能够很好的拟合数据之间的非线性关系，而且 XGBoost 和随机森林均可以并行实现，提升了训练速度。所以我们最终采用 XGBoost 和随机森林模型，通过对比预测的方式，先对 2020 年的五项指标评分进行预测，然后再通过单层神经网络做 LinearRegression 预测总分。

由于评价指标太少，样本数量又相对较多，直接进行模型的训练容易引发欠拟合，导致模型对新增数据的解释力不够。为了降低模型的泛化误差，提高模型的泛化能力，我们采用数据挖掘里的特征工程这一手段，将原始数据转换为更能代表预测模型潜在问题的特征，通过挑选最相关的特征，提取特征以及创造特征。由于待预测数据的特殊性，所以样本特征指标的选取十分重要。

以 SR 指标的预测为例，影响 2020 年 SR 得分的主要因素在于具体的帮扶单位和村庄的属性，不同的单位面对不同的村庄有着不同的办事风格以及积极性，同时 2015 年的包含了 SR 评分的总分排名决定了村庄基础。因此要预测出 2020 年的 SR 评分，就需要知道帮扶单位的编号、被帮扶村庄的属性、2015 年和 2020 年的 SR 评分排名、五年来 SR 评分的排名变化这七项指标信息。为了提高数据质量，我们在确定帮扶单位编号的前提下，针对待预测村庄 2015 年的 SR 评分进行排名，找到同分数的村庄，然后取 2020 年 SR 排名均值作为待预测村庄的 2020 年 SR 名次，然后统计排名变化。同理可以得到其他数据的预测。

5.4.1.2 构建 XGBoost 预测模型预测十个村庄在 2020 年的各指标评分数据

集成算法是在数据上构建多个评估器的模型，然后集成所有模型得到一个综合的结果，提供组合模型的泛化能力，以此来获取比单个模型更好的回归或分类表现。多个模型集成得到的模型叫做集成评估器，组成集成评估器的每个模型都叫做基评估器。一般

有三类集成算法：装袋法（Bagging），提升法（Boosting）和 stacking。装袋法的核心思想是构建多个相互独立的评估器，然后对其预测进行平均或利用多数表决原则来决定集成评估器的结果；提升法的核心思想是结合弱评估器的力量多次对难以评估的样本进行预测，从而构成一个强评估器。

XGBoost 是提升法的典型代表，该算法是基于 GBDT 算法做出的改进[8]，本质是一种提升树模型。它将若干树模型集成在一起进行并行计算，提升训练速度。假设已经训练了 K 颗树，则对于第i个样本的(最终)预测值为:

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i), f_k \in \mathcal{F} \tag{4-1}$$

其中 k 代表第 k 棵树，$f_k(x_i)$代表通过第 k 棵树对样本x_i做预测。将$\sum_{i=1}^{n} l\left(y_i, \hat{y}_i\right)$作为所有样本产生的损失函数，$\sum_{k=1}^{K} \Omega(f_k)$为所有树产生的控制复杂度，可以构建目标函数：

$$Obj = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i\right) + \sum_{k=1}^{K} \Omega(f_k). \tag{4-2}$$

由于该目标函数难以直接优化，因此采用二阶泰勒展开做近似化简得到：

$$\begin{aligned} Obj(t) &= \sum_{i=1}^{n}\left[l\left(y_i, \hat{y}_i^{(k-1)}\right) + \partial_{\hat{y}^{(k-1)}} l\left(y_i, \hat{y}^{(k-1)}\right) * f_k(x_i) + \frac{1}{2}\partial^2_{\hat{y}^{(k-1)}} l\left(y_i, \hat{y}^{(k-1)}\right) * f_k^2(x_i)\right] + \Omega(f_k) \\ &\approx \sum_{i=1}^{n}\left[g_i * f_k(x_i) + \frac{1}{2} h_i * f_k^2(x_i)\right] + \Omega(f_k) \end{aligned} \tag{4-3}$$

其中g_i表示当前叶子结点第 i 个样本的一阶导数$\partial_{\hat{y}^{(k-1)}} l\left(y_i, \hat{y}^{(k-1)}\right)$，$h_i$表示当前叶子结点第 i 个样本的二阶导数$\partial^2_{\hat{y}^{(k-1)}} l\left(y_i, \hat{y}^{(k-1)}\right)$。进一步化简可以得到：

$$Obj = -\frac{1}{2}\sum_{j=1}^{T} \frac{G_j^2}{H_j + \lambda} + \gamma T \tag{4-4}$$

其中G_j表示当前叶子结点所有样本一阶导数的和，H_j表示当前样本所有二阶导数的和，λ为 L2 正则化参数，γ为复杂度的惩罚项。

接下来针对选择的特征计算其所带来的增益，从而选取合适的分裂特征。分支增益公式为：

$$Gain = \frac{1}{2}\left[\frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} + \frac{G_L^2 + G_R^2}{H_L + H_R + \lambda}\right] - \gamma \tag{4-5}$$

搭建 XGBoost 对这十个村庄 2020 年的五项指标评分数据进行预测：

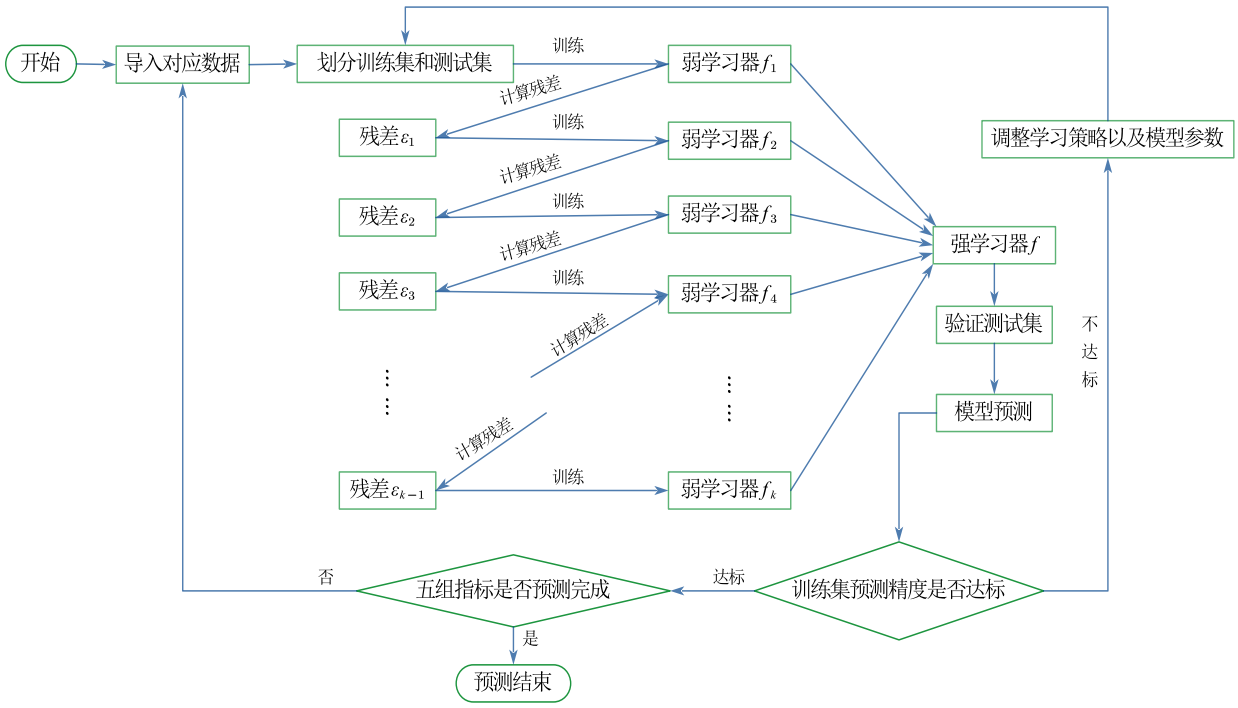

图 4-1　XGBoost 算法流程图

在训练网络之前，需要将原始序列数据划分为测试集和训练集。训练集用来估计模型中的参数，使得模型能够反映现实，进而预测未来或其他未知的信息；测试集用来评估模型的预测性能。首先我们将数据特征及标签进行归一化处理，并随机打乱输入特征，以提高一定的泛化能力。

XGBoost 模型选取的评估指标 RMSE：

$$RMSE = \sqrt{\frac{1}{N}\sum_{t=1}^{N}(observed_t - \text{predicted}_t)^2}$$

基于原始数据以及经特征工程构造的数据，构建 XGBoost 预测模型。为了确定最优的 XGBoost 预测模型，我们将 R^2 作为目标。最终我们确定了最佳预测模型的树的最大深度为 7，弱评估器的个数为 48。

5.4.1.3 构建随机森林预测模型预测十个村庄在 2020 年的各指标评分数据

随机森林是具有代表性的集成算法，其所有基评估器都是决策树。它用随机方式建立一个包含多个决策树的集成分类器，而用回归树所集成的森林就叫随机森林回归器。

我们使用 XGBoost 算法对 2020 年的五项评价指标进行预测，得到的训练模型对于测试数据的 R^2 均在 0.9999 以上。为了检验其预测的可靠性，我们引入随机森林来对 XGBoost 模型预测结果进行对比。由于样本数据量较大，不容易发生过拟合，因此我们选取基尼系数作为不纯度的衡量指标：

$$Gini = 1 - \sum_{i=0}^{n-1}[P(i)]^2 \tag{4-6}$$

建立随机森林模型对十个村庄的 2020 年五项指标进行预测的具体过程如下：

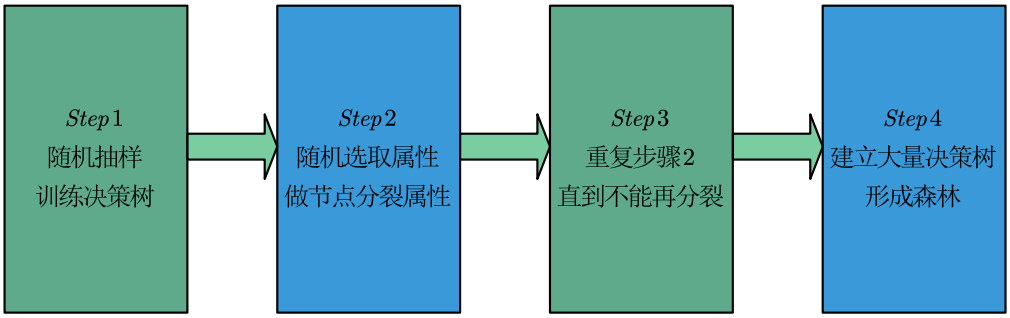

图 4-2 建立随机森林模型预测流程图

5.4.1.4 构建单层神经网络 LinearRegression 模型预测十个村庄的 2020 年总分数据

此前，已将被删除数据的十个村庄的 2020 年五项评价指标的评分数据预测完毕。通过分析原始数据的各项评价指标间的关系发现，评价总分和五项指标评分之间有着密切的线性关系：

$$S_{2020} = 0.1247SR_{2020} + 0.2592CY_{2020} + 0.2466HJ_{2020} + 0.2917WJ_{2020} + 0.2206SS_{2020} \quad (4\text{-}7)$$

因此我们构建单层线性神经网络来对原始数据进行拟合，最后得到的 R^2 为 0.9896200384693424。

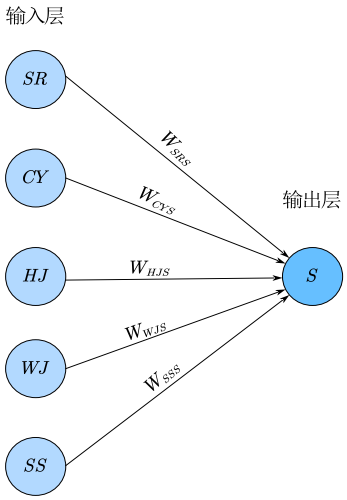

图 4-3 单层线性神经网络结构图

5.4.1.5 预测结果

表 5-1 十个村庄的 2020 年各指标评分数据预测值

单位(村庄)编号	评价指标	XGBOOST		随机森林		两个模型的预测评分之差
		R^2	预测评分	R^2	预测评分	
116 (39257)	SR	0.999988633	1.2877924	0.999952913	1.2878	-7.6E-06
	CY	0.999999164	0.82467186	0.999998319	0.82467	1.86E-06
	HJ	0.999999633	0.94370174	0.999997381	0.94371	-8.26E-06
	WJ	0.9999998	1.0168079	0.999999918	1.0168	7.9E-06
	SS	0.999999461	0.30004793	0.999999296	0.30005	-2.07E-06
…	…	…	…	…	…	…

151 (52436)	SR	0.999988633	-0.17071468	0.999952913	-0.17073	1.532E-05
	CY	0.999999164	-0.3108043	0.999998319	-0.31079	-1.43E-05
	HJ	0.999999633	-0.02508503	0.999997381	-0.02508	-5.03E-06
	WJ	0.9999998	0.11574823	0.999999913	0.11572	2.823E-05
	SS	0.999999461	-0.32856548	0.999999452	-0.32857	4.52E-06

预测结果详见附录

表 5-2 十个村庄的 2020 年各指标评分数据预测值

单位编号	村庄编号	预测总分
116	39257	0.96888313
89	25149	1.20060584
7	12722	0.81496469
10	12916	0.6860089
47	21570	0.52254999
48	22096	-0.0137441
138	47883	0.8289909
78	34208	0.29653316
78	34276	0.68631974
151	52436	-0.14786637

5.4.2 结果分析

影响村庄的各项评价指标得分和评价总分的主要因素是具体的帮扶单位和村庄属性，各单位帮扶的村庄数量不相同，不同单位帮扶不同的村庄有着不同的工作特色，开展工作时的态度、 目标、投入、帮扶干部素质等显然也是有差异的。而被帮扶的村庄原有基础不一致、待开发资源不尽相同也会影响评分。因此我们选取的影响村庄获得“脱贫先进村庄”荣誉称号的重要因素是 2020 年的评价总分排名和五年来各项评价指标得分的进步幅度。是否“先进”主要是依据帮扶村庄各指标评分和总分的进步幅度而不是得分排名，计算出帮扶村庄五年来各项指标评分的变化差之和，变化差越小的说明进步幅度越大。再通过变化差之和排序从 10000 个“脱贫先进村庄”中挑选 2500 个一级“脱贫先进村庄”，再观察我们这十个缺失评分数据的村庄是否在评选范围内。

计划给予 10000 个村庄“脱贫先进村庄”称号，也就是说根据我们选定的评价因素进行“脱贫先进村庄”荣誉称号的评选，排名前 10000 的村庄能够获得荣誉称号。经过观察，这十个村庄中能评上“脱贫先进村庄”的有；

表 5-3 获得“脱贫先进村庄”荣誉称号的村庄

村庄编号	排名
25149	3831
39257	5760
47883	7103
12722	7192
34276	8525
12916	8526

如果称号分为一级和二级（一、二级称号比例为 1:3），那么排名前 2500 名的村庄

能够获得一级荣誉，这十个村庄均不能入选一级“脱贫先进村庄”。

5.5 给国家扶贫办的一封信

尊敬的扶贫办领导：

您好！

当前全国上下都在轰轰烈烈的开展精准扶贫，国家把扶贫开发工作为实现第一个百年奋斗目标的重点工作，并在五年前启动了脱贫帮扶绩效评价机制，以激励各帮扶单位提高扶贫效率，扶真贫，真扶贫。

一个好的脱贫帮扶绩效评价机制能够客观全面地体现出帮扶单位的扶贫成果和贫困村的脱贫提升程度。我们团队运用经过标准化处理的被帮扶村庄集在居民收入、产业发展、居住环境、文化教育、基础设施等五个方面的评分数据以及总分数据，分析得出2015 年与 2020 年的各项指标之间均具有较高正相关性；计算了所有帮扶单位在各单项指标上的帮扶业绩和脱贫帮扶绩效综合得分，并对所有帮扶单位进行绩效、单项指标业绩排序；探索出影响村庄获得“脱贫先进村庄”荣誉称号的重要因素是 2020 年的评价总分排名和五年来各项评价指标得分的进步幅度。

针对我们的研究成果，给出了一些关于脱贫帮扶绩效评价的建议：

（一）避免评定标准的单一化。考虑到扶贫工作中各方面存在的差异，不能仅依靠 2020 年被帮扶村庄的评价总分和各个评价指标得分来确定帮扶单位的绩效名次，同时还要考虑评价总分、各指标评分与五年前相比名次的差异。

（二）要求各单位均衡帮扶所负责的贫困村。应当做到所帮扶的村庄各方面都有均衡提升，防止出现个别村庄明显脱贫提升、某项评价指标得分进步幅度突显等帮扶效果极端化的现象。

（三）充分考虑到帮扶难度不一致对帮扶效率的影响。由于各单位所帮扶的村庄基础、数量都不尽相同，所以开展帮扶工作的难度也不一致，并在一定程度上影响帮扶效率和最终扶贫成果。

我们相信只要公正客观地评价各单位的帮扶效果，就能鼓励更多的帮扶单位愿意花精力投入到扶贫工作中去。

2020 年 8 月 9 日

六、模型的评价

6.1 模型的优点

1、使用斯皮尔曼相关系数探究各项指标的对应关系，可以避免样本数据非正态分布时不能使用皮尔逊相关系数的问题。

2、本文构建的脱贫帮扶评价体系综合考虑了多方面因素对模型的影响，考虑到不同样本的名次提升难度、各项指标的进步幅度和单位帮扶效率，研究的维度和因素较为全面，模型严谨客观。

3、利用因子分析对所有的绩效评定指标进行降维处理，最终提取了三项公共因子，并以其对方差的贡献率为权重加权得到绩效的综合评价指数，较准确地衡量了对不同类型帮扶单位进行评分的差异。

4、使用变异系数来确定 TOPSIS 的指标权重，避免了主观确定权重的不准确性和不稳定性。

5、随机森林可以不用降维就能很好的处理高维度的数据，训练速度快，可以平衡数据集误差；XGBoost 是 GBTD 算法的高效实现，在目标函数中显示的加上了正则化项，考虑了训练数据为稀疏值的情况，可以为缺失值指定分支的默认方向，大大提升了算法的效率。

6.2 模型的缺点

1、利用因子分析法、TOPSIS 法所确定的评定指标不一定能全面覆盖所有因素对帮扶绩效、业绩评价的影响。

2、在对各村庄集在2020年的各项指标评分与总分进行预测时没有利用精准的名次变化。

七、模型的改进与推广

各帮扶单位的脱贫帮扶绩效评价仅利用这五个评价指标提供的信息是远远不够的，应该还应该要添加被帮扶的村庄对帮扶效果的满意程度等等与其他可以体现出扶贫绩效的数据信息。同时还需要获取 32165 个村庄的各评价指标评分的未标准化处理过的数据，让指标评分不只是在空间上与其他对象进行对比，也可以在时间上与自身对比，可以更加明显地体现出帮扶单位扶贫工作开展的成果。绩效评价的多属性决策排序模型可以应用于多领域方面的评价，对不同的对象与方案给出相应的评分和排序。

八、参考文献

[1] 王晓燕, 李美洲. 浅谈等级相关系数与斯皮尔曼等级相关系数[J]. 广东轻工职业技术学院学报, 2006, 5(004):26-27.

[2] 张丽华, 雷琪慧, 蔡林. 基于因子分析法的公立医院运营绩效评价[J]. 卫生软科学, 2019(10).

[3] 金彦, 李奕璋, 郑建. 基于波士顿矩阵的医院绩效管理策略分析[J]. 重庆医学, 2018, 047(015):2098-2100.

[4] 方鹏骞, 张治国, 杨梅. TOPSIS 法在医院绩效评价中的应用[J]. 中国卫生统计, 2005, 22(003):169-170.

[5] 张文朝, 顾雪平. 应用变异系数法和逼近理想解排序法的风电场综合评价[J]. 电网技术, 2014, 38(10):2741-2746.

[6] 孙凯, 鞠晓峰, 李煜华. 基于变异系数法的企业孵化器运行绩效评价[J]. 哈尔滨理工大学学报, 2007, 12(3):165-167.

[7] 郭超. 基于系统动力学的土地整理项目绩效评价[D]. 河北经贸大学, 2015.

[8] Chen, Tianqi, and Carlos Guestrin. “XGBoost.” Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2016): n. pag. Crossref. Web.

九、附录

9.1 问题求解相关表格

表 2-1　因子方差贡献率与累积贡献率

因子	方差百分比	方差累积贡献率
Z_1	52.105	52.105
Z_2	16.278	68.383
Z_3	13.037	81.419
Z_4	6.659	88.078
Z_5	6.182	94.260
Z_6	3.326	97.587
Z_7	2.413	100.000

表 4-1 各项指标评分的预测值

单位编号	评价指标	XGBOOST		随机森林		两个模型的预测评分之差
		R^2	预测评分	R^2	预测评分	
116	SR	0.999988633	1.2877924	0.999952913	1.2878	-7.6E-06
	CY	0.999999164	0.82467186	0.999998319	0.82467	1.86E-06
	HJ	0.999999633	0.94370174	0.999997381	0.94371	-8.26E-06
	WJ	0.9999998	1.0168079	0.999999918	1.0168	7.9E-06
	SS	0.999999461	0.30004793	0.999999296	0.30005	-2.07E-06
89	SR	0.999988633	1.1054943	0.999952913	1.1055	-5.7E-06
	CY	0.999999164	0.78410304	0.999998319	0.78411	-6.96E-06
	HJ	0.999999633	1.154288	0.999997381	1.1543	-1.2E-05
	WJ	0.9999998	1.0988102	0.999999913	1.0987	0.0001102
	SS	0.999999461	1.1571538	0.999999452	1.1573	-0.0001462
7	SR	0.999988633	0.55853784	0.999952913	0.55854	-2.16E-06
	CY	0.999999164	0.54079837	0.999998319	0.5408	-1.63E-06
	HJ	0.999999633	0.8173405	0.999997381	0.81734	5E-07
	WJ	0.9999998	0.77106357	0.999999913	0.77106	3.57E-06
	SS	0.999999461	0.81436545	0.999999452	0.81437	-4.55E-06
10	SR	0.999988633	0.64969814	0.999952913	0.6497	-1.86E-06
	CY	0.999999164	0.2974888	0.999998319	0.29749	-1.2E-06
	HJ	0.999999633	0.9858348	0.999997381	0.98583	4.8E-06
	WJ	0.9999998	0.32050148	0.999999913	0.32051	-8.52E-06
	SS	0.999999461	0.87150407	0.999999452	0.87152	-1.593E-05
47	SR	0.999988633	-0.17071468	0.999952913	-0.17073	1.532E-05

	CY	0.999999164	0.17582613	0.999998319	0.17583	-3.87E-06
	HJ	0.999999633	0.6909797	0.999997381	0.69098	-3E-07
	WJ	0.9999998	0.3819406	0.999999913	0.38195	-9.4E-06
	SS	0.999999461	0.9858183	0.999999452	0.98581	8.3E-06
48	SR	0.999988633	-0.07956928	0.999952913	-0.07957	7.2E-07
	CY	0.999999164	0.4597017	0.999998319	0.4597	1.7E-06
	HJ	0.999999633	-0.15144783	0.999997381	-0.15144	-7.83E-06
	WJ	0.9999998	-0.17098325	0.999999913	-0.17099	6.75E-06
	SS	0.999999461	-0.15708053	0.999999452	-0.15713	4.947E-05
138	SR	0.999988633	0.74084353	0.999952913	0.74085	-6.47E-06
	CY	0.999999164	0.37859318	0.999998319	0.37859	3.18E-06
	HJ	0.999999633	1.154288	0.999997381	1.1543	-1.2E-05
	WJ	0.9999998	0.7300947	0.999999913	0.7301	-5.3E-06
	SS	0.999999461	0.6429261	0.999999452	0.64293	-3.9E-06
138	SR	0.999988633	0.55853784	0.999952913	0.55854	-2.16E-06
	CY	0.999999164	0.50024605	0.999998319	0.50025	-3.95E-06
	HJ	0.999999633	0.60674053	0.999997381	0.60674	5.3E-07
	WJ	0.9999998	-0.27340776	0.999999913	-0.27339	-1.776E-05
	SS	0.999999461	0.12861669	0.999999452	0.12861	6.69E-06
78	SR	0.999988633	0.28506118	0.999952913	0.28506	1.18E-06
	CY	0.999999164	0.9463203	0.999998319	0.94632	3E-07
	HJ	0.999999633	0.7330927	0.999997381	0.7331	-7.3E-06
	WJ	0.9999998	0.525311	0.999999913	0.52531	1E-06
	SS	0.999999461	0.3286187	0.999999452	0.32862	-1.3E-06
151	SR	0.999988633	-0.17071468	0.999952913	-0.17073	1.532E-05
	CY	0.999999164	-0.3108043	0.999998319	-0.31079	-1.43E-05
	HJ	0.999999633	-0.02508503	0.999997381	-0.02508	-5.03E-06
	WJ	0.9999998	0.11574823	0.999999913	0.11572	2.823E-05
	SS	0.999999461	-0.32856548	0.999999452	-0.32857	4.52E-06

9.2 程序代码

Spearman.m

```
clear;clc
%% 程序：计算斯皮尔曼相关系数
%% 导入数据
Test = xlsread('C:\Users\Admin\Desktop\附件 C 题数据.xlsx',2,'N2:W32166');

%% 统计描述
MIN = min(Test);
```

```
MAX = max(Test);
MEAN = mean(Test);
MEDIAN    = median(Test);
SKEWNESS = skewness(Test);
KURTOSIS = kurtosis(Test);
STD = std(Test);
RESULT = [MIN;MAX;MEAN;MEDIAN;SKEWNESS;KURTOSIS;STD]

%% 求解斯皮尔曼相关系数
% 计算矩阵各列的斯皮尔曼相关系数和对应 p 值
[R,P]=corr(Test, 'type' , 'Spearman')
% 显著性符号
P < 0.01    % 3 星
(P < 0.05) .* (P > 0.01)    % 2 星
(P < 0.1) .* (P > 0.05) % 1 星
```

rand_.m	排名		

```
%%
clc;
clear;
%% 导入数据并对原始数据进行降序处理
% save('Rand_data.mat','Rand_data')
load Rand_data.mat;
SR2015 = sort(Rand_data(:,1),'descend');
CY2015 = sort(Rand_data(:,2),'descend');
HJ2015 = sort(Rand_data(:,3),'descend');
WJ2015 = sort(Rand_data(:,4),'descend');
SS2015 = sort(Rand_data(:,5),'descend');
SR2020 = sort(Rand_data(:,6),'descend');
HJ2020 = sort(Rand_data(:,8),'descend');
CY2020 = sort(Rand_data(:,7),'descend');
WJ2020 = sort(Rand_data(:,9),'descend');
SS2020 = sort(Rand_data(:,10),'descend');
% TT2015 = sort(Rand_data(:,11),'descend');
% TT2020 = sort(Rand_data(:,12),'descend');

%% 开始迭代进行 SR2015 排名
%2015
rand_SR = ones(length(SR2015),1);
for k = 2:length(SR2015)
    if SR2015(k-1) == SR2015(k)
```

```
        rand_SR(k) = rand_SR(k-1);
        else
        rand_SR(k) = k;
        end
end
%2020
rand_sr = ones(length(SR2020),1);
for k = 2:length(SR2020)
        if SR2020(k-1) == SR2020(k)
        rand_sr(k) = rand_sr(k-1);
        else
        rand_sr(k) = k;
        end
end
%% 开始迭代进行 CY 排名
%2015
rand_CY = ones(length(CY2015),1);
for k = 2:length(CY2015)
        if CY2015(k-1) == CY2015(k)
        rand_CY(k) = rand_CY(k-1);
        else
        rand_CY(k) = k;
        end
end
%2020
rand_cy = ones(length(CY2020),1);
for k = 2:length(CY2020)
        if CY2020(k-1) == CY2020(k)
        rand_cy(k) = rand_cy(k-1);
        else
        rand_cy(k) = k;
        end
end

%% 开始迭代进行 HJ 排名
%2015
rand_HJ = ones(length(HJ2015),1);
for k = 2:length(HJ2015)
        if HJ2015(k-1) == HJ2015(k)
        rand_HJ(k) = rand_HJ(k-1);
        else
        rand_HJ(k) = k;
```

```
        end
end
%2020
rand_hj = ones(length(HJ2020),1);
for k = 2:length(HJ2020)
        if HJ2020(k-1) == HJ2020(k)
        rand_hj(k) = rand_hj(k-1);
        else
        rand_hj(k) = k;
        end
end

%% 开始迭代进行 WJ2015 排名
%2015
rand_WJ = ones(length(WJ2015),1);
for k = 2:length(WJ2015)
        if WJ2015(k-1) == WJ2015(k)
        rand_WJ(k) = rand_WJ(k-1);
        else
        rand_WJ(k) = k;
        end
end
%2020
rand_wj = ones(length(WJ2020),1);
for k = 2:length(WJ2020)
        if WJ2020(k-1) == WJ2020(k)
        rand_wj(k) = rand_wj(k-1);
        else
        rand_wj(k) = k;
        end
end
%% 开始迭代进行 SS 排名
%2015
rand_SS = ones(length(SS2015),1);
for k = 2:length(SS2015)
        if SS2015(k-1) == SS2015(k)
        rand_SS(k) = rand_SS(k-1);
        else
        rand_SS(k) = k;
        end
end
%2020
```

```
rand_ss = ones(length(SS2020),1);
for k = 2:length(SS2020)
    if SS2020(k-1) == SS2020(k)
    rand_ss(k) = rand_ss(k-1);
    else
    rand_ss(k) = k;
    end
end
%% 开始迭代进行总分 TT 排名
%2015
rand_TT = ones(length(SS2015),1);
for k = 2:length(SS2015)
    if TT2015(k-1) == TT2015(k)
    rand_TT(k) = rand_TT(k-1);
    else
    rand_TT(k) = k;
    end
end
%2020
rand_tt = ones(length(TT2020),1);
for k = 2:length(TT2020)
    if TT2020(k-1) == TT2020(k)
    rand_tt(k) = rand_tt(k-1);
    else
    rand_tt(k) = k;
    end
end
%%
rand = [rand_SR rand_CY rand_WJ rand_HJ rand_SS rand_TT];
```

DEA.m			

```
%%
clc;
clear;
%% 导入数据
Danwei = xlsread('C:\Users\Admin\Desktop\附件 C 题数据.xlsx',4,'A2:A32166');
DEL_SR = xlsread('C:\Users\Admin\Desktop\附件 C 题数据.xlsx',4,'M2:M32166'); %
各个村庄的 SR 名次变动
DEL_CY = xlsread('C:\Users\Admin\Desktop\附件 C 题数据.xlsx',4,'N2:N32166'); %
各个村庄的 CY 名次变动
DEL_HJ = xlsread('C:\Users\Admin\Desktop\附件 C 题数据.xlsx',4,'O2:O32166'); %
```

```
各个村庄的 WJ 名次变动
DEL_WJ = xlsread('C:\Users\Admin\Desktop\附件 C 题数据.xlsx',4,'P2:P32166'); %
各个村庄的 HJ 名次变动
DEL_SS = xlsread('C:\Users\Admin\Desktop\附件 C 题数据.xlsx',4,'Q2:Q32166'); %
各个村庄的 SS 名次变动
DEL_TT = xlsread('C:\Users\Admin\Desktop\附件 C 题数据.xlsx',4,'R2:R32166');
%% 大致统计
DEL_of_mean = (DEL_SR + DEL_CY + DEL_WJ + DEL_HJ + DEL_SS) ./  5; % 各
个村庄的名次变动均值

%% 准备空数组
% 创建空列表
star = 1;
sumSR=[];
sumCY=[];
sumWJ=[];
sumHJ=[];
sumSS=[];
sumTT=[];
n =[];

%% 核心：迭代
for i = 0:159
    sum_SR = 0;
    sum_CY = 0;
    sum_WJ = 0;
    sum_HJ = 0;
    sum_SS = 0;
    sum_TT = 0;
    for k = star:star + 700 %(+800 的目的是巧妙的利用了一个帮扶单位帮了多少
个村子)
        if k>length(Danwei)
            % 这个 break 是针对 Danwei(k) != i 后代表统计下一个帮扶单位对
应数据
            break;
        elseif Danwei(k) == i
            sum_SR = sum_SR + DEL_SR(k);
            sum_CY = sum_CY + DEL_CY(k);
            sum_WJ = sum_WJ + DEL_WJ(k);
            sum_HJ = sum_HJ + DEL_HJ(k);
            sum_SS = sum_SS + DEL_SS(k);
            sum_TT = sum_TT + DEL_TT(k);
```

```
            else
                % 这个 break 是针对 Danwei(k) != i 后代表统计下一个帮扶单位对
应数据
                break;
            end
        end
        n = [n,k - star];
        sumSR = [sumSR,sum_SR]; %各个帮扶单位负责村庄的 SR 排名变动总和
        sumCY = [sumCY,sum_CY]; %各个帮扶单位负责村庄的 CY 排名变动总和
        sumWJ = [sumWJ,sum_WJ]; %各个帮扶单位负责村庄的 WJ 排名变动总和
        sumHJ = [sumHJ,sum_HJ]; %各个帮扶单位负责村庄的 HJ 排名变动总和
        sumSS = [sumSS,sum_SS]; %各个帮扶单位负责村庄的 SS 排名变动总和
        sumTT = [sumTT,sum_TT]; %各个帮扶单位负责村庄的 SS 排名变动总和
        star = k;   %重置统计的起始村庄
end

%% 迭代完成，计算相应指标
% 160 个帮扶单位的各个评价指标对应名次变动均值
mean_of_SR = sumSR./n; %各个帮扶单位负责村庄的 SR 平均排名变动
mean_of_CY = sumCY./n; %各个帮扶单位负责村庄的 CY 平均排名变动
mean_of_WJ = sumWJ./n; %各个帮扶单位负责村庄的 WJ 平均排名变动
mean_of_HJ = sumHJ./n; %各个帮扶单位负责村庄的 HJ 平均排名变动
mean_of_SS = sumSS./n; %各个帮扶单位负责村庄的 SR 平均排名变动
mean_of_TT = sumTT./n; %各个帮扶单位负责村庄的 SR 平均排名变动

Cost = (sumSR - mean_of_SR).^2 + (sumCY - mean_of_CY).^2 + (sumWJ -
mean_of_WJ).^2 +...
        (sumHJ - mean_of_HJ).^2 + (sumSS - mean_of_SS).^2 + (sumTT -
mean_of_TT).^2; %达到均衡水平的代价

DEA_ = 1 ./ (n .*   (sumSR + sumCY + sumWJ + sumHJ + sumSS + sumTT));
zscore_DEA = zscore(DEA_);   %标准化后的 DEA

z_m_SR = zscore([mean_of_SR])';
z_m_CY = zscore([mean_of_CY])';
z_m_WJ = zscore([mean_of_WJ])';
z_m_HJ = zscore([mean_of_HJ])';
z_m_SS = zscore([mean_of_SS])';
z_m_TT = zscore([mean_of_TT])';
z_m_Cost = zscore([Cost])';
z_m_DEA_ = zscore([DEA_])';
z = [z_m_SR,z_m_CY,z_m_WJ,z_m_HJ,z_m_SS,z_m_TT,z_m_Cost,z_m_DEA_];
```

plot_3D.m

```
clf
x = [1:5];
y = x;
[X,Y] = meshgrid(x);
rou = [0.540135843 0.499620092 0.590067303 0.454957696 0.463806496;
0.497142218 0.655629481 0.595351356 0.592826335 0.567074289;
0.584713891 0.607761273 0.800099984 0.550642208 0.560447233;
0.522672683 0.660640169 0.616519402 0.670834283 0.638257622;
0.529287733 0.663192902 0.626570702 0.634082323 0.635794805];
% s = surf(X,Y,rou)
p = plot3(X,Y,rou,'o-');
grid on
xlabel('2020 年')
ylabel('2015 年','rotation',0)
zlabel('斯皮尔曼相关系数')
xticks([1,2,3,4,4.9])
yticks([1:5])

yticklabels({'2015SR','2015CY','2015HJ','2015WJ','2015SS'})
xticklabels({'2020SR','2020CY','2020HJ','2020WJ','2020SS'});
set(gca,'FontSize',12)
```

Guilv.m

```
%%
clc;
clear;
%% 导入数据
rand_SR2015 = xlsread('C:\Users\Admin\Desktop\附件 C 题数据.xlsx',2,'N2:N32166');
rand_SR2020 = xlsread('C:\Users\Admin\Desktop\附件 C 题数据.xlsx',2,'S2:S32166');

%% 对原始数据进行降序处理
% SR2015 = sort(SR2015,'descend');
% SR2020 = sort(SR2020,'descend');
%% 开始迭代进行 SR2015 排名
rand_SR2015 = zeros(length(rand_SR2015),1);
```

```
loar = [ones(3113,1); zeros((length(rand_SR2015)-3113),1)];
rand_SR2015 = rand_SR2015 + loar;
x_10 = find(rand_SR2020 <= 3113);
loar = [ones(6943,1); zeros((length(rand_SR2015)-6943),1)];
rand_SR2015 = rand_SR2015 + loar;
x_20 = find(rand_SR2020 <= 6943);
loar = [ones(9800,1); zeros((length(rand_SR2015)-9800),1)];
rand_SR2015 = rand_SR2015 + loar;
x_30 = find(rand_SR2020 <= 9800);
loar = [ones(12940,1); zeros((length(rand_SR2015)-12940),1)];
rand_SR2015 = rand_SR2015 + loar;
x_40 = find(rand_SR2020 <= 12940);
```

Bianyi.m			

```
clc;
clear;

%% 导入数据
Danwei = xlsread('C:\Users\Admin\Desktop\附件 C 题数据.xlsx','B2:B32166');
DEL_SR = xlsread('C:\Users\Admin\Desktop\附件 C 题数据.xlsx','J2:J32166');
DEL_CY = xlsread('C:\Users\Admin\Desktop\附件 C 题数据.xlsx','K2:K32166');
DEL_HJ = xlsread('C:\Users\Admin\Desktop\附件 C 题数据.xlsx','L2:L32166');
DEL_WJ = xlsread('C:\Users\Admin\Desktop\附件 C 题数据.xlsx','M2:M32166');
DEL_SS = xlsread('C:\Users\Admin\Desktop\附件 C 题数据.xlsx','N2:N32166');
DEL_TT = xlsread('C:\Users\Admin\Desktop\附件 C 题数据.xlsx','O2:O32166');
N1 = N(:,1);
N2 = N(:,2);
N3 = N(:,3);
N4 = N(:,4);
N5 = N(:,5);
%% 准备空数组
% 创建空列表
star = 1;
n =[];
sumN1=[];
sumN2=[];
sumN3=[];
sumN4=[];
sumN5=[];
%% 核心：迭代
```

```
for i = 0:159
    sum_N1 = 0;
    sum_N2 = 0;
    sum_N3 = 0;
    sum_N4 = 0;
    sum_N5 = 0;
    for k = star:star + 700 %(+800 的目的是巧妙的利用了一个帮扶单位帮了多少个村子)
        if k>length(Danwei)
            % 这个 break 是针对 Danwei(k) != i 后代表统计下一个帮扶单位对应数据
            break;
        elseif Danwei(k) == i
            sum_N1 = sum_N1 + N1(k);
            sum_N2 = sum_N2 + N2(k);
            sum_N3 = sum_N3 + N3(k);
            sum_N4 = sum_N4 + N4(k);
            sum_N5 = sum_N5 + N5(k);
        else
            % 这个 break 是针对 Danwei(k) != i 后代表统计下一个帮扶单位对应数据
            break;
        end
    end
    n = [n,k - star];
    sumN1 = [sumN1,sum_N1];
    sumN2 = [sumN2,sum_N2];
    sumN3 = [sumN3,sum_N3];
    sumN4 = [sumN4,sum_N4];
    sumN5 = [sumN5,sum_N5];
    star = k;   %重置统计的起始村庄
end

%% 迭代完成，计算相应指标
% 160 个帮扶单位的各个评价指标对应名次变动均值
mean_of_N1 = sumN1./n;
mean_of_N2 = sumN2./n;
mean_of_N3 = sumN3./n;
mean_of_N4 = sumN4./n;
mean_of_N5 = sumN5./n;

mean_of_N1 = std(sumN1);
```

```
mean_of_N2 = std(sumN2);
mean_of_N3 = std(sumN3);
mean_of_N4 = std(sumN4);
mean_of_N5 = std(sumN5);

mean_of_N1 = mean_of_N1';
mean_of_N2 = mean_of_N2';
mean_of_N3 = mean_of_N3';
mean_of_N4 = mean_of_N4';
mean_of_N5 = mean_of_N5';
MEAN = [mean_of_N1 mean_of_N2 mean_of_N3 mean_of_N4 mean_of_N5]

z_m_N1 = zscore([mean_of_N1])';
z_m_N2 = zscore([mean_of_N2])';
z_m_N3 = zscore([mean_of_N3])';
z_m_N4 = zscore([mean_of_N4])';
z_m_N5 = zscore([mean_of_N5])';
z_m_N = [z_m_N1 z_m_N2 z_m_N3 z_m_N4 z_m_N5];
```

Topsis.m			

```
clear;clc
X = Y(:,[5,10]);
%% 判断是否需要正向化
[n,m] = size(X);
Judge = input(['是否需要经过正向化处理，需要请输入 1 ，不需要输入 0：    ']);
if Judge == 1
    Position = input('请输入需要正向化处理的指标所在的列 ');
    Type = input('  请输入需要处理的这些列的指标类型（1：极小型， 2：中间型，3：区间型）   ');
    for i = 1 : size(Position,2)
        X(:,Position(i)) = Positivization(X(:,Position(i)),Type(i),Position(i));
    end
    disp(' X =   ')
    disp(X)
end

%% 让用户判断是否需要增加权重
Judge = input('请输入是否需要增加权重：  ');
if Judge == 1
    weigh = input(['你需要输入' num2str(m) '个权数。' '请以行向量的形式输入这' num2str(m) '个权重: ']);
```

```
    OK = 0;   % 用来判断用户的输入格式是否正确
    while OK == 0
        if abs(sum(weigh) - 1)<0.000001 && size(weigh,1) == 1 && size(weigh,2)
== m     % 这里要注意浮点数的运算是不精准的。
            OK =1;
        end
    end
else
    weigh = ones(1,m) ./ m ; %如果不需要加权重就默认权重都相同，即都为 1/m
end


%% 对正向化后的矩阵进行标准化
Z = X ./ repmat(sum(X.*X) .^ 0.5, n, 1);
disp('标准化矩阵 Z = ')
disp(Z)

%% 计算与最大值的距离和最小值的距离，并算出得分
D_P = sum([(Z - repmat(max(Z),n,1)) .^ 2 ] .* repmat(weigh,n,1) ,2) .^ 0.5;
D_N = sum([(Z - repmat(min(Z),n,1)) .^ 2 ] .* repmat(weigh,n,1) ,2) .^ 0.5;
S = D_N ./ (D_P+D_N);
disp('最后的得分为：')
stand_S = S / sum(S)
[sorted_S,index] = sort(stand_S ,'descend')
```

Nanduxishu.m			

```
clc;
clear;
SR2015 = xlsread('C:\Users\Admin\Desktop\附件 C 题数据.xlsx',2,'P2:P32166');
CY2015 = xlsread('C:\Users\Admin\Desktop\附件 C 题数据.xlsx',2,'Q2:Q32166');
HJ2015 = xlsread('C:\Users\Admin\Desktop\附件 C 题数据.xlsx',2,'R2:R32166');
WJ2015 = xlsread('C:\Users\Admin\Desktop\附件 C 题数据.xlsx',2,'S2:S32166');
SS2015 = xlsread('C:\Users\Admin\Desktop\附件 C 题数据.xlsx',2,'T2:T32166');
SR2020 = xlsread('C:\Users\Admin\Desktop\附件 C 题数据.xlsx',2,'U2:U32166');
CY2020 = xlsread('C:\Users\Admin\Desktop\附件 C 题数据.xlsx',2,'V2:V32166');
HJ2020 = xlsread('C:\Users\Admin\Desktop\附件 C 题数据.xlsx',2,'W2:W32166');
WJ2020 = xlsread('C:\Users\Admin\Desktop\附件 C 题数据.xlsx',2,'X2:X32166');
SS2020 = xlsread('C:\Users\Admin\Desktop\附件 C 题数据.xlsx',2,'Y2:Y32166');
NSR = zscore(1./(1+exp(-SR2020)) -    1./(1+exp(-SR2015)));
NCY = zscore(1./(1+exp(-CY2020)) -    1./(1+exp(-CY2015)));
NHJ = zscore(1./(1+exp(-HJ2020)) -    1./(1+exp(-HJ2015)));
```

```
NWJ = zscore(1./(1+exp(-WJ2020)) -    1./(1+exp(-WJ2015)));
NSS = zscore(1./(1+exp(-SS2020)) -    1./(1+exp(-SS2015)));

N = [NSR NCY NHJ NWJ NSS];
```

Xgboost 和随机森林.py

```
from sklearn.model_selection import train_test_split as TTS
import pandas as pd
import tensorflow as tf
import numpy as np

''' ================================ 导 入 数 据 ================================ '''
# df = pd.read_csv('./tables/C -  副本.csv', encoding="ISO-8859-1")
df = pd.read_csv('./tables/C 数据.csv', encoding="ISO-8859-1")

''' ================================ 数 据 预 处 理 ================================ '''
X = df.dropna()  # 删除缺失值
t = X

''' ================================ 训 练 测 试 数 据 的 准 备 ================================ '''
# x = np.array([t['Bangfudanwei'],t['bangfutype'], t['2015SR'],t['2015total'],t['2015SR_RAND'],t['2020SR_RAND'],t['del SR']]).T
# y = np.array(t['2020 SR']).reshape(-1, 1)    # 对于一维，必须这样做

# x = np.array([t['Bangfudanwei'],t['bangfutype'], t['2015CY'],t['2015total'],t['2015CY_RAND'],t['2020CY_RAND'],t['del CY']]).T
# y = np.array(t['2020 CY']).reshape(-1, 1)    # 对于一维，必须这样做

x = np.array([t['Bangfudanwei'],t['bangfutype'], t['2015HJ'],t['2015total'],t['2015HJ_RAND'],t['2020HJ_RAND'],t['del HJ']]).T
y = np.array(t['2020 HJ']).reshape(-1, 1)    # 对于一维，必须这样做

# x = np.array([t['Bangfudanwei'],t['bangfutype'], t['2015WJ'],t['2015total'],t['2015WJ_RAND'],t['2020WJ_RAND'],t['del WJ']]).T
# y = np.array(t['2020 WJ']).reshape(-1, 1)    # 对于一维，必须这样做

# x = np.array([t['Bangfudanwei'],t['bangfutype'], t['2015SS'],t['2015total'],t['2015SS_RAND'],t['2020SS_RAND'],t['del SS']]).T
# y = np.array(t['2020 SS']).reshape(-1, 1)    # 对于一维，必须这样做
```

```
# x = StandardScaler().fit_transform(x)
# y = StandardScaler().fit_transform(y)

Xpre = np.array([[116, 1, 1.649, 0.82697, 0.86468, 0.90721, 0.39356, 1.0448, 618, 6775,
4512, 6206, 15840, 2066, 6326, 5790, 5364, 12900, 1448, -449, 1278, -842, -2940],\
                 [89, 1, 1.5766, 1.0095, 0.99283, 1.3535, 1.2463, 1.4405, 821, 3360,
1741, 1, 1, 3133, 7041, 3613, 4518, 4281, 2312, 3681, 1872, 4517, 4280],\
[7, 2, 1.0697, 1.0095, 0.82196, 1.3088, 0.77254, 1.1845, 3726, 3360, 5536, 1347, 8216,
8816, 10034, 7094, 7696, 7468, 5090, 6674, 1558, 6349, -748],\
[10, 2, 1.3593, 0.82697, 0.77925, 1.0857, 1.2463, 1.231, 1671, 6775, 6586, 4420, 1,
8197, 14050, 5503, 12473, 6686, 6526, 7275, -1083, 8053, 6685],\
[47, 1, 0.20065, 0.73572, 1.1637, 1.0411, 0.96204, 1.0448, 15089, 8419, 133, 4420,
4420, 19206, 15814, 8823, 11819, 5670, 4117, 7395, 8690, 7399, 1250],\
[48, 1, -0.01661, 1.0095, 0.01036, 1.0857, 0.39356, 0.63747, 18190, 3360, 21083, 4420,
15840, 18190, 11499, 18353, 17889, 18369, 0, 8139, -2730, 13469, 2529],\
[138, 1, 0.92484, 0.27945, 0.9074, 0.28245, 0.58305, 0.68402, 5181, 17537, 3520,
17491, 12209, 6659, 12768, 3683, 8085, 9078, 1478, -4769, 163, -9406, -3131],\
[78, 1, 0.70758, 0.50758, 0.86468, -0.02993, 0.39356, 0.56764, 7871, 13764, 4512,
21615, 15840, 8626, 10713, 9668, 19083, 14908, 755, -3051, 5156, -2532, -932],\
[78, 1, 0.63516, 1.1007, 0.9074, 0.90721, 0.6778, 1.0332, 8847, 2264, 3520, 6206,
10213, 12923, 4849, 8062, 10167, 12431, 4076, 2585, 4542, 3961, 2218],\
[151, 3, -0.30628, -0.17681, 0.26666, 0.63945, -0.36441, 0.05557, 21779, 23967, 17466,
12209, 25388, 19699, 21512, 17029, 14765, 20300, -2080, -2455, -437, 2556, -5088]])
# xpre = Xpre[:,[0,1,2,7,8,13,18]]
# xpre = Xpre[:,[0,1,3,7,9,14,18]]
xpre = Xpre[:,[0,1,4,7,10,15,18]]
# xpre = Xpre[:,[0,1,5,7,11,16,18]]
# xpre = Xpre[:,[0,1,6,7,12,17,18]]

# Xpre = np.array([[116, 1, 1.649, 0.82697, 0.86468, 0.90721, 0.39356, 1.0448, 618,
6775, 4512, 6206, 15840, 2066, 6326, 5790, 5364, 12900, 1448, -449, 1278, -842, -
2940],\
#                  [89, 1, 1.5766, 1.0095, 0.99283, 1.3535, 1.2463, 1.4405, 821,
3360, 1741, 1, 1, 3133, 7041, 3613, 4518, 4281, 2312, 3681, 1872, 4517, 4280],\
# [7, 2, 1.0697, 1.0095, 0.82196, 1.3088, 0.77254, 1.1845, 3726, 3360, 5536, 1347, 8216,
8816, 10034, 7094, 7696, 7468, 5090, 6674, 1558, 6349, -748],\
# [10, 2, 1.3593, 0.82697, 0.77925, 1.0857, 1.2463, 1.231, 1671, 6775, 6586, 4420, 1,
8197, 14050, 5503, 12473, 6686, 6526, 7275, -1083, 8053, 6685],\
# [47, 1, 0.20065, 0.73572, 1.1637, 1.0411, 0.96204, 1.0448, 15089, 8419, 133, 4420,
4420, 19206, 15814, 8823, 11819, 5670, 4117, 7395, 8690, 7399, 1250],\
# [48, 1, -0.01661, 1.0095, 0.01036, 1.0857, 0.39356, 0.63747, 18190, 3360, 21083,
```

```
4420, 15840, 18190, 11499, 18353, 17889, 18369, 0, 8139, -2730, 13469, 2529],\
# [138, 1, 0.92484, 0.27945, 0.9074, 0.28245, 0.58305, 0.68402, 5181, 17537, 3520,
17491, 12209, 6659, 12768, 3683, 8085, 9078, 1478, -4769, 163, -9406, -3131],\
# [78, 1, 0.70758, 0.50758, 0.86468, -0.02993, 0.39356, 0.56764, 7871, 13764, 4512,
21615, 15840, 8626, 10713, 9668, 19083, 14908, 755, -3051, 5156, -2532, -932],\
# [78, 1, 0.63516, 1.1007, 0.9074, 0.90721, 0.6778, 1.0332, 8847, 2264, 3520, 6206,
10213, 12923, 4849, 8062, 10167, 12431, 4076, 2585, 4542, 3961, 2218],\
# [151, 3, -0.30628, -0.17681, 0.26666, 0.63945, -0.36441, 0.05557, 21779, 23967,
17466, 12209, 25388, 19699, 21512, 17029, 14765, 20300, -2080, -2455, -437, 2556, -
5088]])



model = tf.keras.models.Sequential([
    tf.keras.layers.Dense(128, activation='relu'),
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dense(1, activation='relu')
])

model.compile(optimizer=tf.keras.optimizers.SGD(lr=0.001),
              loss=tf.keras.losses.mse,
              metrics=['accuracy'])
Xtrain, Xtest, Ytrain, Ytest = TTS(x, y, test_size=0.3, random_state=420)  # 划分训练
集测试集
model.fit(Xtrain,Ytrain,  batch_size=32,  epochs=800,  validation_data=(Xtest,
Ytest),validation_freq=1)

model.summary()
```

第四问的回归预测.py			

```
from sklearn.linear_model import LinearRegression as LinearR
from sklearn.model_selection import train_test_split as TTS
import pandas as pd
import numpy as np

'''  ================================  导  入  数  据
================================ '''
# df = pd.read_csv('./tables/C -  副本.csv', encoding="ISO-8859-1")
df = pd.read_csv('./tables/C 数据.csv', encoding="ISO-8859-1")

'''  ================================  数  据  预  处  理
```

```
================================ '''
X = df.dropna()  # 删除缺失值
t = X

'''  ================================  训 练 测 试 数 据 的 准 备
================================ '''
x = np.array([t['2020 SR'],t['2020 CY'], t['2020 HJ'],t['2020 WJ'],t['2020 SS'],]).T
y = np.array(t['2020 total']).reshape(-1, 1)    # 对于一维，必须这样做

# x = StandardScaler().fit_transform(x)
# y = StandardScaler().fit_transform(y)

Xpre = np.array([[1.2877924, 0.82467186, 0.94370174, 1.0168079, 0.30004793],\
[1.1054943, 0.78410304, 1.154288, 1.0988102, 1.1571538],\
[0.55853784, 0.54079837, 0.8173405, 0.77106357, 0.81436545],\
[0.64969814, 0.2974888, 0.9858348, 0.32050148, 0.87150407],\
[-0.17071468, 0.17582613, 0.6909797, 0.3819406, 0.9858183],\
[-0.07956928, 0.4597017, -0.15144783, -0.17098325, -0.15708053],\
[0.74084353, 0.37859318, 1.154288, 0.7300947, 0.6429261],\
[0.55853784, 0.50024605, 0.60674053, -0.27340776, 0.12861669],\
[0.28506118, 0.9463203, 0.7330927, 0.525311, 0.3286187],\
[-0.17071468, -0.3108043, -0.02508503, 0.11574823, -0.32856548]])


Xtrain, Xtest, Ytrain, Ytest = TTS(x, y, test_size=0.3, random_state=420)  # 划分训练集测试集

lr = LinearR()
lr = lr.fit(Xtrain, Ytrain)
print(lr.score(Xtest,Ytest))
# rfc.feature_importances_
print(lr.predict(Xpre))
print(lr.coef_)
```